Adaptive saccade controller inspired by the primates' cerebellum

Marco Antonelli[1], Angel J. Duran[1], Eris Chinellato[2] and Angel P. del Pobil[1]

***Abstract*— Saccades are fast eye movements that allow humans and robots to bring the visual target in the center of the visual field. Saccades are open loop with respect to the vision system, thus their execution require a precise knowledge of the internal model of the oculomotor system. In this work, we modeled the saccade control, taking inspiration from the recurrent loops among the cerebellum and the brainstem. In this model, the brainstem acts as a fixed-inverse model of the oculomotor system, while the cerebellum acts as an adaptive element that learns the internal model of the oculomotor system. The adaptive filter is implemented using a state-of-the-art neural network, called I-SSGPR. The proposed approach, namely *recurrent architecture*, was validated through experiments performed both in simulation and on an antropomorphic robotic head. Moreover, we compared the *recurrent architecture* with another model of the cerebellum, the *feedback error learning*. Achieved results show that the *recurrent architecture* outperforms the *feedback error learning* in terms of accuracy and insensitivity to the choice of the feedback controller.**

I. Introduction

A saccade is a fast and ballistic movement that is used to bring a visual stimulus into the center of the field of view (called fovea in primates). Due to the velocity of the movement, the saccade control is generated without the benefit of sensory feedback. Despite this, the movement is accurate because it takes advantage of an internal model of the visual-oculomotor system [1]. The internal model is plastic and is updated when the parameters of the oculomotor system change [2]. Indeed, once the movement is completed, if there is an incongruity between the observed and the expected position of the target, the internal model is adjusted. In humans, several brain regions are involved in the generation of a saccade [3]. Among these areas, an important role is attributed to the cerebellum which is the area of the brain that stores the internal models of the motor apparatus [4].

From a robotics point of view, generating a saccade requires to solve an inverse control problem, in which the retinotopic position of the stimulus has to be converted into a shift of the eye position. Several techniques have been proposed to solve this problem. Some of them model the inverse controller directly, others employ a forward model or combine the direct and inverse models (see [5] for a review). We adapted two computational models of the cerebellum, the *feedback error learning* (FEL) [6] and the *recurrent architecture* (RA) [7], to address the saccade control problem.

Even though our system is biologically inspired, we limit the parallelism with biological systems to high-level concepts, and we model low-level characteristics according to real-time requirements imposed by robotic applications. Our model takes as inputs the visual target and the current eye position to generate a precise saccade. Both FEL and RA employ a fixed controller and an adaptive element. The fixed controller provides a coarse inverse model of our robot's oculomotor system. The adaptive element implicitly learns the parameters of the system to improve the precision of saccades. It acts as an inverse model in the FEL and as a forward model in RA [7]. The adaptive controller, which represents a high-level model of the cerebellum, is implemented using the incremental sparse spectrum Gaussian process regression (I-SSGPR), a state-of-the-art artificial neural network that guarantees real-time performance, incremental learning and fast convergence [8].

The main analysis about the performance of the controller is conducted by means of exhaustive simulation tests. Finally, the proposed approach is validated on a real humanoid torso.

The remainder of this paper is organized as follows. Section II describes the related work in the fields of robotics and neural networks, while Section III exposes the FEL and the RA. The experimental setup and the achieved results using both simulations and the robot are provided in Section IV. Finally, conclusions are drawn in Section V.

II. Background

A. Inverse controller in robotics

Producing an accurate saccade requires 1) to convert the visual position of the target into a shift of the eye position and 2) to generate an eye movement to get the desired eye position. In our work the eye movement is generated using a PID controller (closed loop with respect to the eye position), so we focus on the transformation that links the visual position of the stimulus into a target position of the eye (open loop with respect to vision, that is visual feed-back is not used during the movement). Learning this transformation requires to learn the inverse kinematic model of the robot head, so we have to cope with the problem of the lack of a teaching signal. Indeed, after a saccade is performed, the adaptive controller would need the motor error to correct the performed movement, but only the visual error is observable. Hence, we need again the inverse model of the system to convert the visual error into a motor error.

*This paper describes research done at the Robotic Intelligence Laboratory. Support for this laboratory is provided in part by Ministerio de Economía y Competitividad (DPI2011-27846), by Generalitat Valenciana PROMETEOII/2014/028 and by Universitat Jaume I (P1-1B2011-54)

[1]M. Antonelli (corresponding author), A.J. Duran and A.P. del Pobil are with Robotic Intelligence Lab, University Jaume I, Spain {antonell,abosch,pobil}@uji.es

[2]E. Chinellato is with School of Computing, University of Leeds, U.K. e.chinellato@leeds.ac.uk

Several strategies have been proposed to solve the inverse control problem. An early proposed strategy is the direct inverse modeling [9]. In its original formulation, the direct inverse modeling consists in performing random movements and then learning the inverse association between the motor command and its perceptual outcome. This technique was employed in the learning of saccade control in several works together with some ad hoc strategies to reduce the exploration process [10], [11], [12]. The main drawbacks of this approach are that it does not cope with redundant systems, and it is not goal-directed. Even if redundancy is not a problem in our application, we still desire a goal-directed approach, in which the system learns while moving toward the observed visual target.

The drawbacks of the direct inverse modeling were addressed by *feedback error learning* (FEL) [6]. FEL consists of two inverse controllers. A fixed feedback controller slowly drives the system toward the target and provides a learning signal to a second adaptive controller. At the beginning the control law is provided entirely by the fixed controller, while, once the system is trained, better performance is obtained due to the effect of the adaptive controller. In robotics, the FEL has been used in several inverse control tasks, among which we can find saccade control [13] and smooth pursuit [14].

Instead of directly learning the inverse controller, the same problem can be addressed by first learning the forward model, and then inverting it by means of an exhaustive incremental searching [15], or by inverting the Jacobian [16]. The inverse and forward models can also be used contextually [17], [18], [19]. For example, the distal teacher approach, uses the inverse of the Jacobian to convert the sensory error into a teaching signal for inverse control [17].

An alternative way to use the forward model is the *recurrent architecture* (RA) proposed by Porrill et al. [20]. A fixed inverse model, such as the one used in the FEL, is combined with a forward model that provides an additional input to the inverse model. The interaction between the inverse-forward model creates a recurrent loop which terminates when the output converges to a stable value. As for the FEL and distal teacher, the adaptive controller is learned incrementally during goal directed movements. However, in this case, the teaching signal is provided directly by the sensory error obtained as output of the plant. Thus, the teaching signal is proximal to the sensory processing. Moreover, it does not require the inversion of the Jacobian, which is not a biologically plausible solution. The RA was proposed as a computational model of the cerebellum, and it was tested in several simulations [20], [7], [21] and in one robotic setup to learn the vestibulo-oculomotor control [22] but not for saccade movements.

B. Neural networks and visuomotor transformations

Learning the inverse model, as described in the previous section, requires an adaptive controller that can be tuned according to the input-output data. In this section we focus on the neural networks that have only one hidden layer. The reason is twofold. The first one is related to the modelling of the cerebellum, that suggests that one hidden layer (granular cells) provides the basis functions that are linearly combined to compute the output activation (Purkinje cells) [21]. The second reason is that one-layer neural networks can be trained using fast converging algorithms [23], [24], which is a desirable property in robotic applications.

In previous works we addressed the reference frame transformation problem by using Gaussian basis function networks [25], [26], [27], [28]. One drawback of this approach is that, due to the gradient descent based learning rule, the networks converge slowly. Moreover, they suffer the curse of dimensionality: the number of neural units increases geometrically with the number of input variables.

One way to address the curse of dimensionality is provided by the *growing neural networks*, that increase the number of neurons depending on the error of the approximation. An example of this approach, which was tested in the learning of saccade control [13], is the dynamic cell structure [29].

The problem of speeding up the convergence rate can be solved using the recursive least square [23] or the Kalman filter [24]. These approaches update the covariance of the weights in order to combine efficiently new and old observations. Using these techniques, in our previous works we learned eye-hand coordination with three degrees of freedom [30], [31] and we exploited the covariance matrix to choose the most informative visual targets for learning the inverse saccade control [32].

The advantages provided by the recursive least square and by the growing neural networks were combined in the local weighted projective regression (LWPR) [33]. These networks have been very successful in several applications in robotics also for generating the eye trajectory necessary to execute a saccade [14], due to their capacity of learning with high degrees of freedom.

However, LWPR are quite difficult to use due to the number of parameters that need to be set. An alternative to the use of growing neural networks is to approximate the Gaussian activation of neurons using sparse features [34]. This approach has been used in a recent neural network called incremental sparse spectrum Gaussian process regression (I-SSGPR) [8]. Beside handling high dimensionality input, the I-SSGPR updates the weights using an incremental algorithm that performs the *maximum a posteriori* estimate. This algorithm has very few parameters, that can be tuned using log marginal likelihood optimization [8]. In this work, we use the I-SSGPR to implement the adaptive controller that is required to generate precise saccades.

III. Model

This section describes the two architectures that we have implemented for the generation of the saccade control, the FEL and the RA. The goal of both models is to convert the visual target $\vec{t}$ and the current eye position $\vec{e}$ into an eye shift (saccade) $\Delta\vec{e}$, that would bring the stimulus into the center of the visual field. Among possible alternatives for representing visual information we favor the composition of a cyclopean image representation (c_x, c_y) with a disparity

map (d), over the option of having separate left (u_l, v_l) and right (u_r, v_r) representations:

$$\vec{t} = \begin{bmatrix} c_y \\ c_x \\ d \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_l \\ u_r \\ v_l \\ v_r \end{bmatrix} \quad (1)$$

We represent the gaze direction by means of the tilt (θ_t), version (θ_{vs}) and vergence (θ_{vg}) angles:

$$\vec{e} = \begin{bmatrix} \theta_t \\ \theta_{vs} \\ \theta_{vg} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} \theta_t \\ \theta_l \\ \theta_r \end{bmatrix} \quad (2)$$

where θ_l and θ_r are angular position of the left and right eye, respectively.

Both architectures are composed by a fixed controller $\mathbf{B}$ and a non-linear adaptive controller $\mathbf{C}(\cdot)$. In this work we set $\mathbf{B}$ to be a linear inverse model of the oculomotor system $\mathbf{P}(\cdot)$ (plant). Even if using $\mathbf{B}$ alone is enough to drive the eyes toward the target, the execution of a ballistic movement does not provide a precise saccade. This is due to the non-linearity of the oculomotor system, which is not modelled correctly by the linear controller. In order to improve its performance, we add an adaptive controller $\mathbf{C}(\cdot)$. The difference between the two control schemes is how the controllers $\mathbf{B}$ and $\mathbf{C}(\cdot)$ are interconnected, and the role of the adaptive controller $\mathbf{C}(\cdot)$, which is an inverse model ($\mathbf{C_f}(\cdot)$) in the FEL and a forward model ($\mathbf{C_r}(\cdot)$) in the RA. The details of the two approaches are provided in the following sections.

Feedback Error Learning

In the FEL the adaptive controller provides an inverse model of the plant which is used to correct the output of $\mathbf{B}$. The input of the controller is the visual target $\vec{t}$ and the current eye position $\vec{e}$ while the output is the saccade command $\Delta\vec{e}$. The eye movement is then sent to the robotic head which moves accordingly. After the movement, the new visual position of the stimulus $\vec{t}'$ is converted into a motor error to adapt the inverse controller $\mathbf{C_f}(\cdot)$. This conversion is performed again by the approximated inverse model $\mathbf{B}$ (see Fig. 1). Using this approach, the adaptive filter learns to compensate the poor response of the fixed feedback control, and the expected response of the cerebellum is $\mathbf{C}_f = \mathbf{P}(\cdot)^{-1} - \mathbf{B}$ [7].

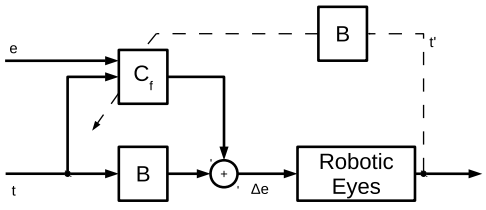

Fig. 1. Feedback error learning. The visual position of the stimulus ($\vec{t}$) and the current eye positions ($\vec{e}$) are converted into a motor command ($\Delta\vec{e}$) by summing up the contributions of a fixed element $\mathbf{B}$ and an adaptive element $\mathbf{C_f}(\cdot)$. Training the weights of $\mathbf{C_f}(\cdot)$ requires a motor error $\mathbf{P}(t')^{-1} \approx \mathbf{B} \times \vec{t}'$ instead of a sensory error $\vec{t}'$.

Recurrent Architecture

In the RA we have the same linear inverse model ($\mathbf{B}$) that we used in the FEL, but in this case the adaptive controller $\mathbf{C_r}(\cdot)$ provides a correction of the input that is sent to the linear controller $\mathbf{B}$. This correction is fed into the linear controller to obtain a new eye shift. Using the new command, the adaptive controller provides a new correction, thus creating a loop between the fixed and the adaptive elements. This loop terminates when the motor command converges to a stable value. In our experiments we set a threshold on the increment of the motor command, and we also limit the maximum number of iterations. In this configuration, the input of the adaptive controller are the visual target $\vec{t}$, the current gaze position $\vec{e}$ and the motor command $\Delta\vec{e}$. One of the main advantages of using this approach is that the teaching signal is provided by the sensory error $\vec{t}'$, without needing an additional transformation as in the case of FEL (see figure 2). Using this approach, the adaptive controller converge to $\mathbf{C}_r(\cdot) \approx \mathbf{B}^{-1} - \mathbf{P}(\cdot)$ [7].

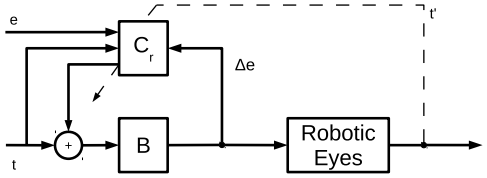

Fig. 2. Recurrent architecture. The visual stimulus is sent to the fixed element $\mathbf{B}$ that generates a motor command $\Delta\vec{e}$. This command, together with the visual target $\vec{t}$ and the current eye position $\vec{e}$, provides the input to the adaptive element $\mathbf{C_r}(\cdot)$. The output of $\mathbf{C_r}(\cdot)$ is then used as a correction to the input to $\mathbf{B}$. In this recurrent architecture, the visual stimulus obtained after the movement of the head is used directly as a teaching signal.

Implementation

Both control schemes were composed by fixed and adaptive controllers. In the fields of computational neuroscience and bio-inspired robotics, the role of the fixed controller is usually associated to the brainstem, while the adaptive component is associated to the cerebellum [35], [13]. The fixed controller provides a fixed linear inverse model of the plant so that: $\Delta\vec{e} \approx \mathbf{B} \times \vec{t}$. In our application, $\mathbf{B}$ is a 3×3 matrix that was calculated using the least square method on the input-output data of the dataset. The matrix $\mathbf{B}$ is the a priori information that we have about the system, thus we prefer an architecture that is insensitive to the choice of $\mathbf{B}$, as long as it represents a reasonable approximation of the inverse model of the plant.

The adaptive controllers were different for the two architectures, but both were implemented using an I-SSGPR network with 500 random features composed of sine and cosine functions (see [8] for details). In FEL, $\mathbf{C_f}(\cdot)$ has a six-dimensional input composed of the visual target ($\vec{t}_b$) and the current eyes' position ($\vec{e}_b$), while the output is the eyes' movement that should correct the output provided by

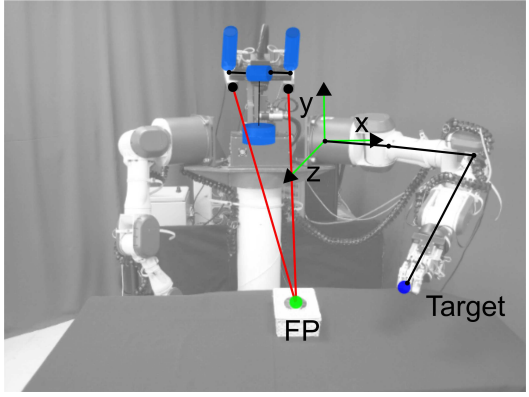

Fig. 3. The UJI (University Jaume I) humanoid torso: Tombatossals. Blue cyclinders represent the four joints of the head. FP: fixation point.

$\mathbf{B}$ alone. In the RA, $\mathbf{C_r}(\cdot)$ has a nine-dimensional input composed of the target ($\vec{t}_b$), the current eyes position ($\vec{e}_b$) and the upcoming eyes movement ($\Delta\vec{e}_b$), while the output is the correction of the input which is sent to $\mathbf{B}$.

In the FEL, the linear model is used to convert the visual error into a motor signal (distal error) which is used to train the network (see Fig. 1). This makes the teaching signal (estimated motor error) depending on to choice of $\mathbf{B}$. This does not happen in the RA, in which the visual error directly provides the teaching signal (proximal error) for the adaptive controller (see Fig. 2). Therefore, we expect FEL to be more sensitive to the choice of $\mathbf{B}$ with respect to the RA.

IV. Experiments and Results

A. Robot setup

Our robot *Tombatossals* is a humanoid torso endowed with a mechatronic head and two multi-joint arms (Fig. 3). The robotic head (*Robosoft TO40*) mounts two cameras with a resolution of 1024×768 pixels that can acquire color images at 30 Hz (*The Imaging Source, DFK 31AF03-Z2*). The pixel size is $4.65\ \mu m$ and the focal length was set to $5\ mm$ to obtain a wide field of view. The cameras can actively move by means of a common pan (not used in this work), a common tilt (θ_t) and two independent pan motor (θ_l and θ_l). The four joints of the robotic head are shown in Fig. 3.

The baseline between the cameras is about $270\ mm$. The center of rotation of the motors does not lie on the optical center of the camera lens, so that their rotation produces both rotation and translation of the optical point. Due to this translational component, the visuo-oculomotor transformation depends on the distance from the target. This misalignment between the optical center and the center of rotation depends on the focal length of the camera, and it is present virtually in every robotic system and also in the human eyes [36]. In our setup we measured a displacement of about $48\ mm$ between optical center and center of rotation.

B. Methodology and Simulation Results

In this section we describe the experiment that evaluates the proposed architectures FEL and RA using a 3D simulation of the kinematics of the robot described in the previous section.

Methodology: The dataset was composed of 8205 points acquired with the following protocol. We placed 500 virtual targets in the space in front of the robot, and we defined 125 eye positions in the gaze-centered frame of reference. For each eye position we computed the projection of the 500 stimuli in the left (u_l, v_l) and right images (u_r, v_r). Then, for each target that was observable in both images at the same time, we stored in the dataset the eye position and the visual stimulus. The dataset covered a wide region in the peripersonal space of the robot, as can be seen in Table I, showing the range and the distribution of the sample points.

TABLE I

Range and distribution of the input data.

Input	Min. Value	Max. value	Mean	Std. Dev.
c_y pixels	-382.6	372.6	-1.4	180.91
c_x pixels	-503.4	507.7	7.3	240.8
d pixels	-560.4	625.1	-17.1	204.9
θ_t degrees	-70.25	70.25	-0.33	23.91
θ_{vs} degrees	-70.21	70.22	-0.50	24.62
θ_{vg} degress	3.50	38.19	17.39	8.87

In each experiment we trained the neural networks adopting an on-line strategy that can be replicated on the robot. For each sample of the training, we computed the eye shift ($\Delta\vec{e}$) by multiplying the visual position of the stimulus ($\vec{t}$) by matrix $\mathbf{B}$. The achieved eye shift was then corrected using the adaptive controller $\mathbf{C}(\cdot)$. In the case of the FEL, the controller $\mathbf{C_f}(\cdot)$ was fed with the visual target and the current eye position ($\vec{e}$). In the case of the recurrent loop, the adaptive controller $\mathbf{C_r}(\cdot)$ obtained as input also the output of the linear controller ($\Delta\vec{e}$). The recurrent loop terminated when the correcting factor calculated by the controller was lower than 1 pixel, or when it reached 30 iterations. In both architectures we initialized the weights of the adaptive controller to zero, so at the beginning they did not influence the inverse control. Once the eyes' movement was computed, we simulated the execution of a saccade and then we computed the projection of the target on both cameras. The new visual position of the target ($\vec{t}'$) was used to change the weights of the adaptive controller. The testing phase followed the same paradigm of the training phase, without adapting the networks. In this case, we also used the configuration of the eyes after the saccade to compute the fixation point of the robot in Cartesian space.

We trained and tested the two architectures using the K-Fold cross validation, with K=5. The error of a saccade was calculated as the Euclidean distance of the stimulus from the center of the image.

Results: In order to study the sensitivity of the system to this matrix, in the experiments we multiplied the linear controller $\mathbf{B}$ by a scalar gain that was changed systematically between 0.5 and 1.4. Fig. 4 shows the mean and the standard deviation of the error on the test set for the two architectures, as a function of the gain used by the linear controller.

Using the FEL the best performance was achieved with the gain set to 1.2. In this condition the mean error was 1.07 ± 1.57 pixels. Using this gain, in the worst trial the

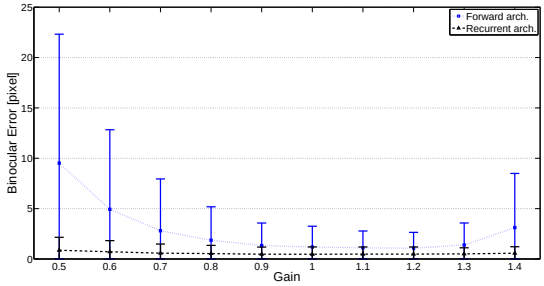

Fig. 4. Performance of FEL and RA as a function of the gain used to change the fixed linear controller. Markers represent the mean error of the saccade, bars represent one standard deviation.

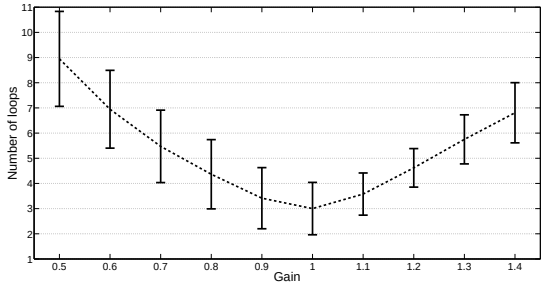

Fig. 5. Number of loops (mean and standard deviation) in the *recurrent architecture* as a function of the gain.

saccade ended at 27.37 pixels, not very far from the center, considering that the size of the image is 1024×768 pixels. In the RA, the best performance was achieved with a gain set to 1.0. In this condition the mean error was 0.47 ± 0.72 pixels. In the worst trial the saccade ended at 25.15 pixels from the center. Thus, in the worst case the two architectures provide similar results, but the average behavior of the RA is twice as good as the FEL, even if the input dimension is bigger and the neural networks have the same number of units (500). The difference between the two approaches is significant when the gain changes. Indeed, for FEL, the performance degraded considerably with low and high gains. For example, with a gain of 0.5 the mean error was 9.5 ± 12.80 pixels and the worst trial left the target at a distance of 235.55 pixels from the center. With the same gain, the mean error of the RA was 0.86 ± 1.29 pixels and the worst trial left the target at a distance of 31.66 pixels from the center (see Fig. 4). So, the performance of RA is more invariant to the choice of the gain, and this solution is preferable for saccade control.

A disadvantage of the RA is the computational time required to compute the eyes' movement. As shown in Fig. 5, the number of iterations depends on the chosen gain. With the tested gains, the average number of loops is between 3 ± 1 and 9 ± 2, while the maximum number of loops changes from 10 to 24. Thus, the computation time of the RA depends on the trial, and it is not stationary as in the case of the FEL. In any case, the computational time required to compute 10 loops is lower than $1ms$ on a $Intel(R), Core(TM)\, i5 - 2520M$, fast enough to guarantee real time performance.

C. Robot experiments

After the analysis in simulation, we validated some results on our humanoid torso (Fig. 3).

Methodology: The visual processing on the robot was simplified by using as a target a red label placed on the fingertip of the robot left hand. The arm of the robot was used to position the target in the Cartesian space[1].

For both control architectures, we trained the network from scratch using the same paradigm used during the simulations. In this case, we placed the target (fingertip) into 125 positions of the Cartesian space. For each target position the eyes started the saccadic behavior from 26 different gazing directions. From each position of theeyes, the robot performed a saccade toward the target and then used the post-saccadic visual error to train the network. In this way we acquired 3250 training points.

In a second phase, we tested the algorithm on other points of the visual space. In this case we initialize the gaze of the robot to 1000 random positions and the task was to perform a ballistic eye movement toward the visual target. After each saccade we recorded the Euclidean distance of the target from the center of the cyclopean visual field. In this case we skipped the training step.

Results: Using a gain set to 1, the average radial error on the test set was 4.32 ± 2.34 pixels for the FEL and 3.70 ± 2.75 pixels for the RA. Using a gain of 0.5 we achieved a result of 6.47 ± 4.2 pixels using the FEL (performance decrease of 50 percent) and 4.83 ± 3.40 pixels using the RA (performance decrease of 31 percent). The improvement provided by the RA with respect to the FEL is less evident than it is in the simulations results, probably because we tested the algorithm on a smaller region of the space. However, these results are in accordance with the ones obtained in simulation, and show that the RA is less sensitive to the choice of $\mathbf{B}$ and it is slightly more accurate.

TABLE II

PERFORMANCE OF THE SACCADIC CONTROL IN THE LITERATURE.

Approach	Resolution	Error [pixel]	Error [μm]
Recurrent architecture	1024×768	3.70 ± 2.75	17.2
Feedback error learning	1024×768	4.32 ± 2.34	20.1
Forssén[15]	640×480	5.5	40.7
Bruske[13]	512×512	2.5	27

Table II provides an overview of the results reported in the related literature. Forssén et al. [15] employed a forward model that was inverted using exhaustive incremental searching, while Bruske et al. [13] employed the FEL using dynamic cell structure [29]. To compare the results we converted the error provided by the authors into an error in μm using the pixel size of the cameras (third and fourth columns in Table II). Results show that our implementation of the FEL using I-SSGPR outperforms the state-of-the-art approaches and, more importantly, RA is better than the FEL in the same experimental conditions (first two lines in

[1]The origin of this frame is centered on the shoulder of the robot, as shown in Fig. 3.

Table II).

V. Conclusion

This work is part of our research on a sensorimotor framework that aims at creating an implicit representation of space based on visual and somatosensory cues [37]. Here we focused on the association between the retinotopic encoding of the target and its eye-centered representation. The internal model that describes this association is learned by means of saccadic eye movements which in turn help to create the representation of the surrounding space. On the other hand, accurate saccades are executed thanks to the recurrent interaction between a fixed feedback controller, that emulates the brainstem, and the adaptive internal model, that emulates the cerebellum. The internal model is maintained by an adaptive filter implemented using state-of-the-art neural network, namely I-SSGPR. Experimental results show that our implementation of the saccade control ouperforms results achieved in the literature. Moreover, we compared the *recurrent architecture* (RA) with the *feedback error learning* (FEL) and we conclude that the RA is more accurate and less sensitive to the choice of the inverse model with respect to FEL. Future work is directed toward the study of how the knowledge of our internal model can influence the attentive process in order to speed-up the learning process.

References

[1] H. Chen-Harris, W. Joiner, V. Ethier, D. Zee, and R. Shadmehr, "Adaptive control of saccades via internal feedback," *The Journal of Neuroscience*, vol. 28, no. 11, p. 2804, 2008.

[2] M. Lappe, "What is adapted in saccadic adaptation?" *The Journal of Physiology*, vol. 587, no. 1, pp. 5–5, Oct 2008.

[3] B. Girard and A. Berthoz, "From brainstem to cortex: computational models of saccade generation circuitry," *Progress in Neurobiology*, vol. 77, no. 4, pp. 215–251, 2005.

[4] D. M. Wolpert, R. C. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends Cog Sci*, vol. 2, no. 9, pp. 338–347, 1998.

[5] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[6] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," *Advanced neural computers*, vol. 6, no. 3, pp. 365–372, 1990.

[7] J. Porrill and P. Dean, "Recurrent cerebellar loops simplify adaptive control of redundant and nonlinear motor systems," *Neural computation*, vol. 19, no. 1, pp. 170–193, 2007.

[8] A. Gijsberts and G. Metta, "Real-time model learning using incremental sparse spectrum gaussian process regression," *Neural Netw*, 2012.

[9] M. Kuperstein, "Neural model of adaptive hand-eye coordination for single postures," *Science*, vol. 239, pp. 1308–1311, 1988.

[10] W. Schenck and R. Möller, "Learning strategies for saccade control," *Künstliche Intelligenz*, no. 3/06, pp. 19–22, 2006.

[11] F. Chao, M. Lee, and J. Lee, "A developmental algorithm for ocular-motor coordination," *Robotics and Autonomous Systems*, vol. 58, no. 3, pp. 239–248, 2010.

[12] M. Antonelli, E. Chinellato, and A. P. del Pobil, *On-Line Learning of the Visuomotor Transformations on a Humanoid Robot*, ser. Advances in Intelligent Systems and Computing. Springer Berlin Heidelberg, 2013, vol. 193, pp. 853–861.

[13] J. Bruske, M. Hansen, L. Riehn, and G. Sommer, "Biologically inspired calibration-free adaptive saccade control of a binocular camera-head," *Biological Cybernetics*, vol. 77, no. 6, pp. 433–446, 1997.

[14] T. Shibata, S. Vijayakumar, J. Conradt, and S. Schaal, "Biomimetic oculomotor control," *Adapt. Behav.*, vol. 9, no. 3-4, pp. 189–207, Apr. 2001.

[15] P. Forssén, "Learning saccadic gaze control via motion prediciton," *Computer and Robot Vision (CRV), 2007. Fourth Canadian Conference on*, pp. 44–54, 2007.

[16] G. Sun and B. Scassellati, "Reaching through learned forward model," *Humanoid Robots, IEEE/RAS International Conference on*, vol. 1, pp. 93– 112 Vol. 1, 2004.

[17] M. Jordan and D. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive Science: A Multidisciplinary Journal*, vol. 16, no. 3, pp. 307–354, 1992.

[18] D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Netw*, vol. 11, no. 7, pp. 1317–1329, 1998.

[19] B. Damas, L. Jamone, and J. Santos-Victor, "Open and closed-loop task space trajectory control of redundant robots using learned models," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Nov 2013, pp. 163–169.

[20] J. Porrill, P. Dean, and J. V. Stone, "Recurrent cerebellar architecture solves the motor-error problem," *Proceedings of the Royal Society of London-B*, vol. 271, no. 1541, pp. 789–796, 2004.

[21] J. Porrill, P. Dean, and S. R. Anderson, "Adaptive filters and internal models: Multilevel description of cerebellar function," *Neural Networks*, vol. 47, pp. 134–149, 2013.

[22] A. Lenz, T. Balakrishnan, A. G. Pipe, and C. Melhuish, "An adaptive gaze stabilization controller inspired by the vestibulo-ocular reflex," *Bioinspiration & biomimetics*, vol. 3, no. 3, p. 035001, 2008.

[23] N. B. Karayiannis and A. N. Venetsanopoulos, "Fast learning algorithms for neural networks," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 7, pp. 1–22, Jul 1992.

[24] S. S. Haykin *et al.*, *Kalman filtering and neural networks*. Wiley Online Library, 2001.

[25] E. Chinellato, M. Antonelli, B. J. Grzyb, and A. P. del Pobil, "Implicit sensorimotor mapping of the peripersonal space by gazing and reaching," *IEEE Trans. Auton. Mental Develop*, vol. 3, pp. 43–53, 2011.

[26] M. Antonelli, E. Chinellato, and A. P. del Pobil, "Implicit mapping of the peripersonal space of a humanoid robot," in *Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2011 IEEE Symposium on*, 2011.

[27] E. Chinellato, M. Antonelli, and A. P. del Pobil, "A pilot study on saccadic adaptation experiments with robots," in *Biomimetic and Biohybrid Systems*, TJ Prescott et al., Ed. Springer Berlin Heidelberg, 2012, vol. 7375, pp. 83–94.

[28] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural computation*, vol. 3, no. 2, pp. 246–257, 1991.

[29] J. Bruske and G. Sommer, "Dynamic cell structure learns perfectly topology preserving map," *Neural Computation*, vol. 7, no. 4, pp. 845–865, 1995.

[30] M. Antonelli, B. J. Grzyb, V. Castelló, and A. P. del Pobil, *Plastic Representation of the Reachable Space for a Humanoid Robot*, ser. LNCS. Springer Berlin Heidelberg, 2012, vol. 7426, pp. 167–176.

[31] M. Antonelli, B. Grzyb, V. Castelló, and A. del Pobil, "Augmenting the reachable space in the nao humanoid robot," in *Workshop at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[32] M. Antonelli, A. Duran, E. Chinellato, and A. P. del Pobil, "Speeding-up the learning of saccade control," in *Biomimetic and Biohybrid Systems*, ser. LNCS, N.F. Lepora et al., Ed. Springer Berlin Heidelberg, 2013, vol. 8064, pp. 12–23.

[33] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[34] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in neural information processing systems*, 2007, pp. 1177–1184.

[35] P. Dean, J. E. Mayhew, and P. Langdon, "Learning and maintaining saccadic accuracy: a model of brainstem–cerebellar interactions," *Journal of Cognitive Neuroscience*, vol. 6, no. 2, pp. 117–138, 1994.

[36] F. Santini and M. Rucci, "Active estimation of distance in a robotic system that replicates human eye movement," *Robotics and Autonomous Systems*, vol. 55, no. 2, pp. 107–121, 2007.

[37] M. Antonelli, A. Gibaldi, F. Beuth, A. J. Duran, A. Canessa, M. Chessa, F. Solari, A. del Pobil, F. Hamker, E. Chinellato, and S. Sabatini, "A hierarchical system for a distributed representation of the peripersonal space of a humanoid robot," *IEEE Trans. Auton. Mental Develop*, pp. 1–15, 2014.